RIFT Working Group
Internet-Draft
Intended status: Informational
Expires: December 15, 2018

Y. Filyurin, Ed.
Bloomberg LP

June 13, 2018

RIFT -- Motivation, Additional Requirements and Use Cases in User Access Networks

draft-filyurin-rift-access-networks-00

Abstract

RIFT is a new specialized dynamic routing protocol originally designed for Clos and Fat Tree Data Center networks. It is designed to work on multilevel network topologies in which nodes in certain level will only connect to nodes in one upper or lower level with optional and non-contiguous intra-level connectivity.

While the protocol was originally designed to meet the needs of Massively Scalable Data Centers, its ability to automatically prune the information distribution from higher levels to lower levels, as well as provide optimal routing for intra and inter-level traffic makes it a good match for user access networks, or any network that combines end user access and various compute enabling various network service for these end users. Current directions in distributed computing seek to blur even that distinction. Large distributed networks can be created, where virtual compute units can be in all tiers, combining and crossing many requirements for DC or User Access design. This draft seeks to analyze these requirements.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 15, 2018.

Table of Contents

1. Definitions of Terms Used in This Memo

MSDC - Massively Scalable Data Center

IGP - Interior Gateway Protocol

RIB - Routing Information Base

FIB - Forwarding Information Base

MT - Mutli-Topology in the context of IS-IS

MI - Mutli-Instance in the context of IS-IS

AD - Auto-discovery

UDP - User Datagram Protocol

IID - Instance ID, in the context of control and data plane slicing of network devices

TIE - Topology Information Element, per original RIFT specification

N-TIE - Northbound Topology Information Element, flooded in the Northbound direction, per original RIFT specification

S-TIE - Northbound Topology Information Element, propagated in the Southbound direction, per original RIFT specification

Node TIE - Node Topology Information Element, per original RIFT specification

Prefix TIE - Prefix Topology Information Element, per original RIFT specification

Key Value TIE or K/V TIE - A TIE (mainly Southbound) that is carrying a set of key value pairs, per original RIFT specification

LIE - Link Information Element, per original RIFT specification

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] when, and only when, they appear in all capitals, as shown here.

2. Authors

Following authors substantially contributed to the current format of the document:

3. Introduction

Typical access networks are built in a hierarchical fashion using "Core", "Distribution" and "Access" layers designed to support collections of wiring distribution blocks that in turn connect to end user devices, server compute nodes and various forms of utility devices. This design is just variation of the Fat Tree design and RIFT presents an opportunity to significantly reduce traditional switched networks design limitations, bring seamless mobility to end systems within the entire access network domain and remove the operational overhead that comes with provisioning access networks. All this can be done without forcing lower level network devices to carry feature sets traditionally found in higher end aggregation devices.

Decoupling network layer information from device reachability information allows any network layer information to be propagated, and thus, expand the protocol to support routing for any type of network layer addressing. Use of Policy Guided Prefixes allows specialized forwarding policies where packets are forwarded through specialized paths or redirected to specialized service nodes, such as packet shapers. Use of Key/Value N-TIEs and S-TIEs would allow propagation of both configuration information to facilitate fully automated deployment and operations. Key/Value TIEs can be used to propagate other information that can aid forwarding such as interface queuing policies, access control policies or configuration of auxiliary services such as DHCP relay. The use of IPv6 Link Local addressing on all infrastructure for exchange of LIEs removes a lot of operational overhead in bringing up and supporting RIFT network.

4. Additional Requirements for RIFT Access Networks

The original RIFT specification was created for traditional Data Center environments. Access networks may call for additional capabilities. This desire for additional capabilities is due to the fact that many endpoints in these traditional access environments often lack the capabilities of providing traditional delineation between the network infrastructure domain and individual workloads running on these devices and must rely on the network edge to provide that delineation.

5. Network Slicing

Network slicing in this context is defined as creating individual separate virtual networks within our access networks connecting sets of edge devices. The slices are effectively their own virtual Fat Trees with separate Control Plane data structures holding prefix information. The protocol processes populate virtual RIBs, which

program the FIB (assuming common FIB in most platforms) to define instance specific packet identification and its per hop forwarding behavior.

Network slices can also be called network instances. Often they are used interchangeably, but often network instance applies a virtual network construct local to an individual device, where network slice covers a virtual network carved out from the set of interconnected devices.

5.1. Overall Network Slicing

RIFT original specification uses the concepts of Multi-Topology RFC5120 [1] and Multi-Instance RFC6822 [2] to create network-wide virtual routing domains. RIFT capabilities to form separate neighbor relationship for each instance make MI approach more appropriate for creating network slices, allowing multiple virtual Fat Trees to operate as "ships in the night" creating completely separate RIFT flooding/propagation domains. As part of initial LIE exchange individual adjacencies per instance will be formed, as long as the nodes can agree on the instance ID. Standard discovery process can apply, and it could be argued that all auto-configuration information exchange can happen only at the global instance.

5.2. Identification and Propagation of Slice Information

The process is no different in principle than for many other forms of virtual private network services. The process starts with Auto-Discovery where nodes hosting a particular instance can propagate this information to other nodes (using Key/Value (K/V) Ties, for example) and individual neighbor relationships will form. Once instance adjacencies form, then all other information can be exchanged and propagated.

Since RIFT is fundamentally an underlay protocol, and relies on itself for next hop resolution, instance awareness must not just be on edge devices hosting the instance, but all transit devices. In both MT and MI approaches, topologies and instances are explicitly configured. When provisioning RIFT networks, there must be some approach to facilitate instance activation on transit devices. Once the device becomes "instance aware", then LIE exchange can take place to establish common parameters such as UDP ports and neighbor adjacency can be established using standard process.

Due to K/V capabilities of RIFT, there should be no need to define special Instance ID TLVs or modify the Thrift models. Some external entity will configure instance parameters and access policies

instance system IDs established and K/V N-TIEs can be propagated to higher levels and neighbor adjacencies established.

5.3. Network Instances and RIB and FIB Requirement

While RIFT is an underlay protocol, as soon as individual virtual Fat Trees are created, packet forwarding on links, that are used for multiple slices can no longer be programmed using standard network layer information. This is a typical example of using some unique identifier to determine unique per-hop behavior. The price of using unique identifiers whether they take on the form of shim headers, special packet metadata or even translation and encapsulation techniques is the requirement of creating more advanced forwarding state on the transit network devices. First, there must be an association that maps a particular identifier to a particular instance, then another action that makes the forwarding decision identifying the next hop and the final action of adding the right metadata to the packet allowing the next hop to perform the same set of actions.

The problem can be resolved using two standard approaches outside of deploying multi-operation forwarding devices. Either put the destination address based forwarding on the edges, that already have the policies to associate network layer information with instance information, or create more advanced FIB data structures that map to hardware operations that allow metadata/address lookup and forwarding to be done as simple atomic operations.

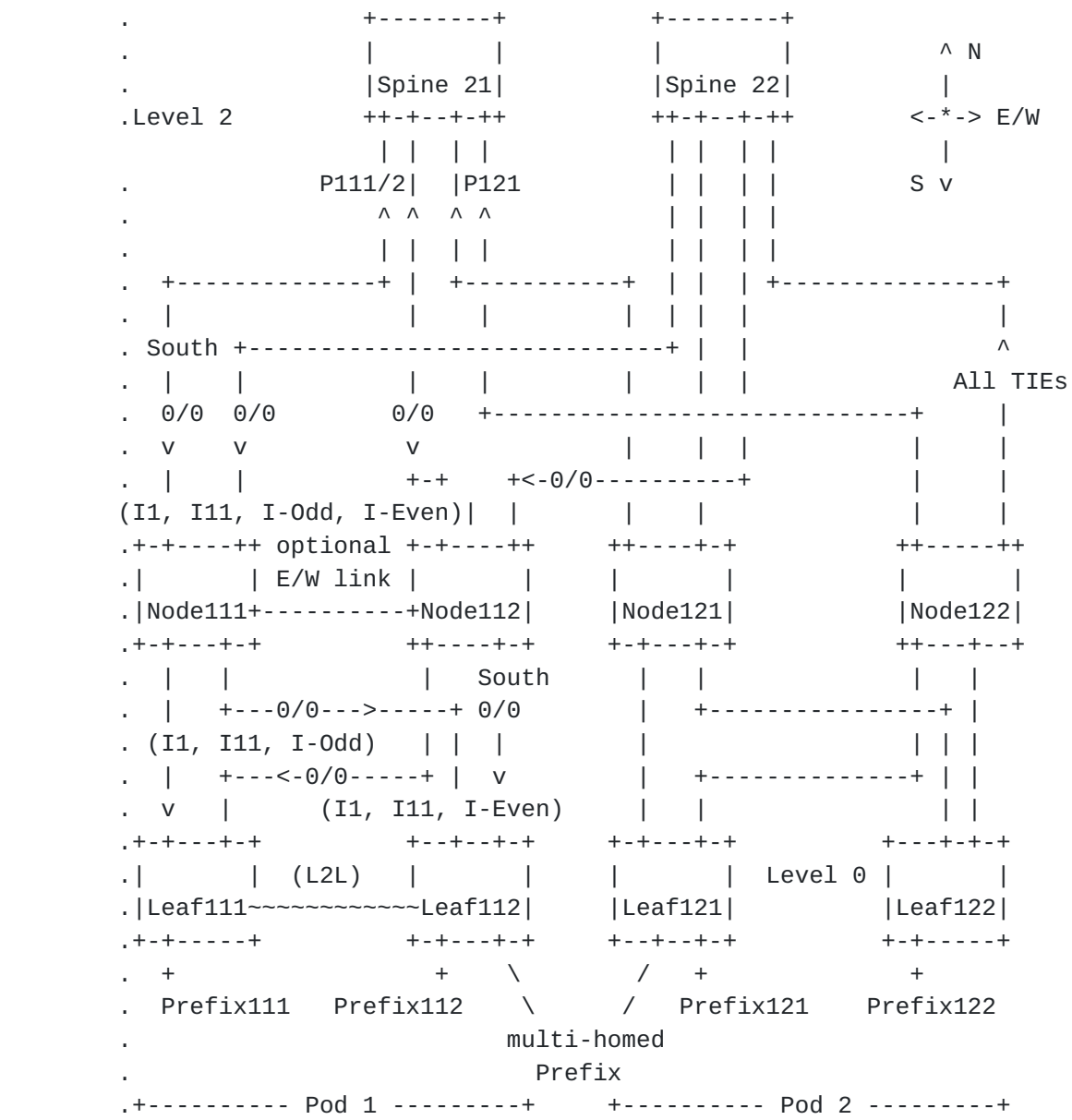
The first approach to some degree defeats the purpose of using RIFT as a routing protocol - access devices having visibility to all destinations and metadata available to them. The second approach is more realistic, but these advanced capabilities may only be available on more advanced devices. These devices are less likely to be deployed closer to edges of RIFT network, and possibly get in the way of the requirement of less expensive and feature rich access network.

Within MI RIFT domain, there would be three types of forwarding behavior. First forwarding behavior is on the leaf devices connecting to endpoints that apply policies associating end systems with instances, impose and dispose of the metadata and forward the packet to transit devices. The second forwarding behavior is found on transit devices. These devices forward exclusively based on metadata, or shim headers, effectively forwarding traffic to the highest level aggregation devices. The last type of device is the aggregation device, that maintains advanced FIB that processes and forwards packets, imposing metadata used to forward to the leaf devices.

5.4. Network Instances and Control Plane

Taking the example drawing from original RIFT spec:

:



A two level spine-and-leaf topology

Assuming we take every "Leaf" device (111,112,121 and 122) and create instance I1 on each device, as well as instance policies. At the same time, Leafs 111 and 112 can host an instance I11 and leafs 121 and 122 can host instance I12. 111 and 121 are hosting I-Odd and 112 and 122 are hosting I-even. Northbound K/V TIEs can be used to propagate instance information and set up instance RIB data structures on the transit devices. Leafs will have those data structures set up during instance creation and transit devices as soon as they receive K/V TIEs. In this example Spines will have the RIB data structures for all the instances created, Node 111 and Node 112 should only have the state from I1, I11 and I-Odd and I-Even and Nodes 121 and 122 should have the identical state, except that I11 would be replaced by I12.

The same approach would be applied to forming adjacencies. Once the initial LIE exchange completes and instance TIEs have been exchanged between the devices and parameter negotiation is complete - instance specific neighbor adjacency can be established. The creation of all the data structures, TIE flooding and propagation starts then.

In the above setup, the leafs maintain the needed Control Plane state created as part of configuration and propagation of Prefix S-TIEs from transit nodes. Their 0/0 or ::/0 or any other relevant routing state within each instance is designed to route packets towards the spines.

Transit nodes (Node 111, 112, 121 and 122) would have instance adjacencies with leafs based on which leaf hosts which instance. For example all transit nodes with maintain I1 adjacency with every leaf, but I-Even adjacency with leafs 112 and 122 and I-Odd with 111 and 112. Since per instance adjacencies are formed this is even more flexible than MI-ISIS, and there is no need to do IID TLV mechanism. A direct association exists between instance RIB data structures and per instance adjacencies.

Spines 22 and 22 would create RIB data structures for all the instances, as the spines are responsible for routing the traffic between leafs. In our example their adjacencies are still based on advertise K/V TIEs indicating instance memberships. Spines 21 and 22 would have all the instance adjacencies with nodes 111 and 112, except for instance I12 and with 121 and 122 for all the instances, except for I11.

All the standard RIFT rules must apply for adjacency establishment on horizontal links between nodes of the same level. The same rules must apply for prefix disaggregation and treatment of Policy Group Prefix (PGP) TIEs.

A leaf is expected to connect to multiple nodes and failure of instance synchronization on the horizontal link either indicates and outage of an error. It could be up to the implementation to define default behavior and correlation of K/V TIEs with flooded Node TIEs

5.5. Network Instances and Forwarding

Leafs apply instance policies, dispose of the metadata and make forwarding decisions to forward packets to spines through various node transit devices. This is the primary difference between normal RIFT operation and per-instance RIFT, designed to address the forwarding limitations of transit devices, that would have to identify the topology, perform the forwarding action within the context of that topology and potentially put another identifier for the next device. Leafs therefore must not just forward the packets, but impose the right information on it, to allow transparent forwarding to the spines by transit devices. Spines in turn have the task of identifying the topology, determining the leaf device for the destination address (for any address schema) and properly marking the packet for topology identification as it is forwarded towards the destination leaf.

Techniques for forwarding packets to the spines and then to the appropriate leafs can be up to implementations or may be hardware specific, where some set-ups are better off with encapsulation, some better of with shim headers and some with address manipulation. The forwarding tables of transit devices must have the information to forward packets to the spines, and multiple instances can share that information, as long as spines can uniquely identify the instance.

This is a potential use case for various techniques ranging from simple Label Switched Paths (LSPs) using both label swapping and forwarding, more complex approaches for path set-up with use of deeper label stacks to identify the devices, instance and some other per-hop behavior. Doing this conflicts with the Requirement #13 as outlined in the original RIFT draft, where all traffic must transit the spine. This may be very much acceptable in a traditional user access network, where most of the traffic is ultimately North/South or has to be North/South due to various security requirements, but as traffic patterns change, various systems become more distributed and enterprise data processing starts resembling smaller scale MSDCs, it may not be a bad idea to have the capability to have multiple levels of devices capable of executing advanced per-hop actions on the packets.

6. External Routing Information

In most environments RIFT will not be the only control plane protocol. Recent advances in compute virtualization designs create an opportunity for designs in which traditional compute hosts are now running multiple workloads where network virtualization is now at the network layer, as opposed to traditional approach of transport layer virtualization. As such, individual virtual operating system instances or virtual processes present their own network layer address. These addresses exist on the network only for the duration of the workload and in some situations even move. This applies to primarily Data Center networks and while these can found in access environments, the scale requirements are unlikely to be significant. In access environments, however, server compute nodes are replaced by numerous systems that in turn support mobile devices, special purpose mesh networks.

Mobility will be discussed later, but various control plane protocols can be deployed on lower level nodes, especially leaf nodes, where these external protocols are used to create routing information used to forward packet to these compute workloads. In addition to these protocols, Network Admission Control protocols as well as network discovery protocols can be used to populate device routing tables.

All this routing information must be exchanged with RIFT as part of export/import relationship between RIFT and RIB manager or redistribution between RIFT and databases of these protocols. These foreign prefixes are propagated as Prefix TIEs Northbound with the ability to carry some information that identifies these as external and some additional information allowing non-leaf devices to treat the information in a special way. Prefix TIEs are able to carry optional attribute set. As part of this optional set, Route Tags can be defined and used for external route identification. Aside from the optional attribute set, there would not even be difference between "internal" and "external" prefixes, as the import process is nearly identical.

External routes by default should not be propagated Southbound and would be subject of the same de-aggregation rules that apply to normal RIFT operation. External prefixes would only be propagated southbound if the node in the southern direction could follow the default in the direction where there would no visibility of that route. Implementation should offer the option to propagate external routes without any explicit configuration. Situations, in which RIFT domain could be used to interconnect other routing domains can be a match for this requirement.

RIFT is not meant to become an inter-domain routing protocol, but various forms of stub networks of many compute and transit entities using other specialized routing protocols could be interconnected using RIFT domain, as well as connecting to other external systems.

7. RIFT and Endpoint Address Mobility

Most of endpoint addressing including network addressing belongs to fixed locations, as the network address is associated with a connecting interface. When service endpoints have their own addresses that exist independent of network addresses, this separation ultimately creates the need for address mobility. Endpoint address mobility is both the ability to move the association of any address endpoint to any network device interface, as well as ability to reuse any endpoint address anywhere in the mobility domain.

Numerous traditional approaches exist ranging from relying on combining locator and endpoints in a single address, keeping all endpoints in a continuous broadcast domain relying on auto-discovery mechanisms to various centralized and distributed Locator/Endpoint mapping systems, that keep track of endpoint mobility. Numerous work went into making both approaches scalable utilizing various networking layers, but the problem has been reduced to one of distributed dynamic routing - ability to re-advertise the address of the endpoint to reroute to it through a different locator.

7.1. Mobility Use Cases

Actual mobility use cases may include activation, deactivation and moves of virtual compute systems in both server and access environment. They can be both virtual servers serving clients to virtual nodes in various peer-to-peer applications. Other use cases may include activation, deactivation and association of wireless nodes to different point-to-point, point-to-multipoint and mesh wireless networks. Whether the locator is a physical compute node or a wireless access point, the locator serves as the boundary between the static locator and dynamic endpoint networks. RIFT takes on dynamically routing in the first to support access to the second.

RIFT support for mobility is defined in the Mobility section of the RIFT specification. The fundamental requirement for the mobile node management systems, whether centralized or distributed is to support notifying RIFT either through redistribution/import mechanism or directly when mobility events happen, as RIFT does not have a native purge mechanism and RIFT will insure the right network state to provide routing to the right locator is maintained using time stamp

and sequence counter mechanisms. Both unicast and unicast routing can be supported.

Address mobility should be supported in both single global instances as well as multi-instance configuration. For non-global instances RIFT operation should be no different, and each instance would maintain its own data structures keeping track of timestamps and sequence numbers. As stated in the original RIFT specification, mobility can be defined as a service and supported through a separate instance. If done so, then various transit nodes between leafs and super-spines are either forwarding encapsulated packets or programmed to process just shim headers and metadata. While this does not minimize the control plane effort needed to perform mobility at scale (as RIFT is an underlay protocol), this would reduce the FIB sizes and minimize the data plane requirements.

As outlined in the original RIFT specifications, some environments would already be designed to support mobility using other techniques for locator/endpoint separation such as LISP or ILA. While RIFT can assist these protocols with providing the needed configuration to the leaf nodes, such as instance mapping and resolver information, the two systems operate independently.

8. Border Nodes and Superspine East/West traffic

Border Nodes are special purpose leaf nodes connected directly to the top level of the hierarchy. They may run a foreign routing protocol and will often be used to interconnect to different networks. Most of the external routes, including the default routes would be originated from those. The first approach is to treat these devices as any other nodes in the hierarchy. They will assume a lower level and will flood N-TIEs and receive standard Node S-TIEs and all Prefix S-TIEs. They are just regular leafs, but because of their function, they are capable of propagating external routing information and also receive all prefix TIEs, as opposed to just originated default.

The second approach is to have the mechanism to treat the super-spine, other interconnected super-spines and border nodes (which become super-spines at this point) as part of a single flood domain. This is similar as treating super-spines as a traditional backbone area in OSPF or Layer 2 domain in IS-IS. All N-TIEs are flooded on all links in the higher available level.

This is a request to have the only allowed exception to the original specification that explicitly states that neither N-SPF nor S-SPF can provide full loop prevention capability as the entire Fat Tree design is not based on the continuous connectivity at any level. If the super-spine domain becomes its own Link State flooding domain, or

East/West TIEs are introduced, than Prefix S-TIEs must be used to populate the RIB if they are available. In addition East/West ties can never be used to propagate information as S-TIEs. Super-spines do not get to act as backbone areas, or various techniques used for things like route leaking have to be employed.

9. Border Nodes and Superspine East/West traffic

Where super-spines represent the top of the hierarchy bringing various design ideas and their caveats such as continuous super-spine domain, leafs also want to take advantage of certain topology optimizations. In certain set-ups especially in large campus and metro area networks, leaf connectivity can be deployed in a "daisy chain" fashion. In such connectivity set-up, a set of leaf devices will be interconnected where the "leftmost" and "rightmost" devices provide connectivity to higher levels of the tree. Similar to the interconnected super-spine concept, this violates some of the design principles of Fat Tree topology and some accommodations for this in the RIFT protocol may be required.

RIFT is not designed to provide full ring protection, unless the ring consists of 2-3 nodes (becoming either an interconnected single tier or leaf/spine with a single leaf). A ring of more than 3 nodes becomes a broken Fat Tree topology. Before a multilevel RIFT environment with the bottom level being a daisy chain of leafs, we can try a simple ring approach. Assuming two adjacent nodes on the ring can be configured as SUPER_SPINE, then it is theoretically possible that all other nodes of the newly formed "half-ring" could have a level assigned to them, and depending on the number of nodes in a ring, one or two nodes would become level 0 leafs. Assuming that we would want all the nodes to become leafs, then either the nodes must be explicitly configured to be LEAF_ONLY, or the links from the two "aggregation nodes" to the leaf nodes must be configured to explicitly tell other nodes that they are leafs, which those leaf nodes must continue propagating. This may require creation of another flag used in adjacency formation.

Assuming the correct adjacencies have been formed and we have a set of two nodes: Node1 and Node2 of level 1 and a set of leafs, Leaf1, Leaf2 and Leaf3 where Leaf1 connects to Node1 and Leaf2, Leaf2 connects to Leaf1 and Leaf3 and Leaf3 connects to Node2. Nodes 1 and Node 2 can either have a direct E/W link or just links to Nodes of Level 2.

The first design violation is breaking one of the Leaf-to-Leaf rules, which states that only the N-TIEs that are originated by a particular leaf are sent over East/West Leaf-to-Leaf link. Since the leaf devices in a daisy chain are part of the same level, this rule could

be relaxed, as N-TIEs from leafs in the chain can be propagated to higher levels where they get to run N-SPF and deal with partitioned leaf network. The condition of this relaxation can be that devices in the daisy chain ultimately rely on S-SPF only based on what is propagated with S-TIEs. S-TIEs in turn get propagated in both directions of the chain without being sent Northbound.

Endpoints on devices in the half-ring rely on S-TIEs to reach other endpoints in this sub-topology and S-TIEs to reach endpoints outside the half ring. N-TIEs are propagated to allow endpoints outside the half-ring to reach endpoints in the half-ring. Partition within the half-ring would have to trigger the reflooding of the N-TIEs, as well as propagation of the S-TIEs. This may be the only possible situation on which a purge like Southbound mechanism is used, but ultimately the direction is not Southbound, but East-West.

10. Security Considerations

Access environments are less trusted environments. RIFT is designed in such a way to make it possible for a device to join the network without too much extra configuration. The protocol was designed to simplify operations, but at the price at making it a lot easier for devices to become part of the network. In many MSDC environments the devices are deployed to come online with special interfaces that connect to dedicated Out-of-Band (OOB) management network. Not only the process preconfigures these devices, the system ensures that the initial configuration as well as software and firmware of the version that a particular enterprise considers secure. Devices deployed in more remote locations or just those without out of band management network connectivity may not go through the initial configuration, plus general physical security is lowered. Security procedures for neighbor authentication become a lot more critical.

Implementing Secure Neighbor Discovery would make the attachment to the network more difficult and implementing a protocol that supports encryption could keep protocol communications secure.

A number of activities in 6lo working group have developed a number of ideas that could create a more secure way for the RIFT neighbors to authenticate each before forming a formation. Of note is the work done in RFC6775 [3] as well as its extension that addresses security

11. Conclusions

RIFT started out as a Data Center protocol, and will evolve in that direction, allowing greater scalability in building multi-tier fabrics. As the requirements of MSDCs and larger access environments start to look very similar, and as end user compute and server

compute start performing very similar functions, there will be more similarities between end user mobility and workload mobility than there differences. RIFT and its enhancements that combine many aspects of control and management planes, can become the IGP these environments have been waiting for.

12. IANA Considerations

At this point there is no need for any allocations

13. Acknowledgments

Would like to acknowledge Antoni Przygienda for the original feedback and hopefully steering this document in the right direction.

14. References

14.1. Normative References

[I-D.ietf-rift-rift]
Przygienda, T., Sharma, A., Thubert, P., Atlas, A., and J. Drake, "RIFT: Routing in Fat Trees", draft-ietf-rift-rift-01 (work in progress), April 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

14.2. Informative References

[RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <https://www.rfc-editor.org/info/rfc3971>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <https://www.rfc-editor.org/info/rfc5120>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC8202] Ginsberg, L., Previdi, S., and W. Henderickx, "IS-IS Multi-Instance", RFC 8202, DOI 10.17487/RFC8202, June 2017, <https://www.rfc-editor.org/info/rfc8202>.

14.3. URIs

[1] https://tools.ietf.org/html/rfc5120

[2] https://tools.ietf.org/html/rfc6822

[3] https://tools.ietf.org/html/rfc6775

Author's Address

Yan Filyurin (editor)
Bloomberg LP
731 Lexington Ave.
New York, NY 10022
US

EMail: yfilyurin@bloomberg.net